**CSS Selectors & Styling**

**Q.1 : What is a CSS selector? Provide examples of element, class, and ID selectors.**

**Ans.** CSS Selector: A pattern used to select HTML elements to apply styles.

Three types of selector:

1. TAG NAME
2. CLASS NAME
3. ID NAME

**Q.2: Explain the concept of CSS specificity. How do conflicts between multiple styles get resolved?**

**Ans.** CSS Specificity:

CSS specificity determines which style rule is applied when multiple rules target the same element. It’s calculated based on the type of selectors:

* Inline styles: highest priority (e.g., style="")
* ID selectors (#id): strong
* Class, attribute, pseudo-class selectors (.class, [type], :hover): medium
* Element and pseudo-element selectors (div, ::before): weak

Conflict Resolution:

1. Higher specificity wins.
2. If specificity is equal, the later rule (in the code) wins.
3. important overrides all, regardless of specificity (but should be used sparingly).

**Q.3 What is the difference between internal, external, and inline CSS? Discuss the advantages and disadvantages of each approach.**

**Ans.**

**Inline CSS:-**

* **Definition:** Styles written inside the HTML element using style attribute.
* **Advantage:** Quick, good for single elements.
* **Disadvantage:** Not reusable, messy for large projects.

**Internal CSS:-**

* **Definition:** Styles written inside <style> tag in the HTML <head>.
* **Advantage:** Easy for single-page styling, no extra file needed.
* **Disadvantage:** Works only for that page, harder to maintain for big sites.

**External CSS:-**

* **Definition:** Styles written in a separate. css file and linked to HTML.
* **Advantage:** Reusable across pages, clean, best for large projects.
* **Disadvantage:** Needs extra file loading; if file fails, no styles applied.

In the short: **Inline = one element,**

**Internal = one page,**

**External = many page.**

**CSS Box Model**

Q.1 Explain the CSS box model and its components (content, padding, border, margin). How does each affect the size of an element?

Ans.

**CSS Box Model:-**

Every element in CSS is a box with 4 parts:

1. **Content** – actual text or image inside.
2. **Padding** – space between content and border.
3. **Border** – line around the padding/content.
4. **Margin** – space outside the border, separating elements.

**Effect on size:**

* Content, padding, and border **increase the element’s size**.
* Margin adds **space around** the element (doesn’t increase content size).

Q.2 What is the difference between border-box and content-box box-sizing in CSS? Which is the default?

Ans.

**Difference between content-box and border-box:**

1. **content-box (default)**

* width and height apply **only to content**.
* Padding and border are **added outside**, increasing total size.

1. **border-box**

* width and height include **content + padding + border**.
* Total size stays fixed, easier for layouts.

**CSS Flexbox**

Q.1  What is CSS Flexbox, and how is it useful for layout design? Explain the terms flex-container and flex-item.

Ans.

**CSS Flexbox:-**

* **Flexbox (Flexible Box Layout):** A CSS layout model that makes it easy to design **responsive** and **aligned layouts** without using floats or complex positioning.
* It automatically adjusts elements’ size and space to fit different screen sizes.

**Key Terms:-**

1. **Flex-Container**
   * The parent element where display: flex; is applied.
   * It controls how child elements are arranged (direction, alignment, spacing).
2. **Flex-Item**
   * The child elements inside a flex-container.
   * They are the boxes that get aligned, resized, and distributed within the container.

**In the short:** Flexbox = easy, responsive layout.

* **Flex-container** = parent.
* **Flex-item** = children inside.

Q.2 Describe the properties justify-content, align-items, and flex-direction used in Flexbox.

Ans.

* **flex-direction** → Defines the direction of flex items (row, row-reverse, column, column-reverse).
* **justify-content** → Aligns items along the **main axis** (start, end, center, space-between, space-around, space-evenly).
* **align-items** → Aligns items along the **cross axis** (stretch, start, end, center, baseline).

In the short:  
flex-direction sets the axis, justify-content aligns along the axis, and align-items aligns across the axis.

**CSS Grid**

Q.1 Explain CSS Grid and how it differs from Flexbox. When would you use Grid over Flexbox?

Ans. **CSS Grid vs Flexbox**

* **CSS Grid** is a **two-dimensional layout system** (rows + columns). It’s best for page layouts, image galleries, or dashboards where you need precise control of both horizontal and vertical placement.
* **Flexbox** is a **one-dimensional layout system** (row *or* column). It’s best for navbars, menus, toolbars, or aligning items in a single line.

**Key difference:**

* Grid = good for full layouts (structure first).
* Flexbox = good for arranging items in a line (content first).

**Use Grid when:** you need structured rows & columns.  
**Use Flexbox when:** you need flexible alignment along one axis.

Q.2 Describe the grid-template-columns, grid-template-rows, and grid-gap properties. Provide examples of how to use them.

Ans. 1. **grid-template-columns** → sets number & width of columns.  
*Example:* grid-template-columns: 200px 1fr 100px;

**2. grid-template-rows** → sets number & height of rows.  
*Example:* grid-template-rows: 100px auto 50px;

**3.grid-gap** (or gap) → sets space between rows & columns.  
 *Example:* grid-gap: 20px;